SHRIMATI INDIRA GANDHI COLLEGE

Affiliated to Bharathidasan University| Nationally Accredited at 'A' Grade(3rd Cycle) by NAAC

An ISO 9001:2015 Certified Institution

Thiruchirrappalli

STUDY MATERIAL

SOFTWARE ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE, INFORMATION TECHNOLOGY AND COMPUTER APPLICATIONS

Prepared by,

MS. C. SHYAMALADEVI, M.C.A., M.Phil., B.Ed.,

ASST. PROF. IN COMPUTER SCIENCE,

SHRIMATI INDIRA GANDHI COLLEGE,

TIRUCHIRAPPALLI – 2

Software Maintenance

Software maintenance is a part of the Software Development Life Cycle. Its primary goal is to modify and update software application after delivery to correct errors and to improve performance. Software is a model of the real world. When the real world changes, the software require alteration wherever possible.

Software Product: Is the result of software development

Software Maintenance: Results in a service being delivered to the customer

The difference between the development and maintenance which will help us to understand the maintenance-related activities in detail is listed below.

Development	Maintenance
Application is not into the production environment	Application is already into the production environment
End user is not using the system	End user started using the system
It is not driven by the end users. It is driven by the project team	It is completely driven by the end users
Opportunity for innovation is more	Opportunity for innovation is less
Time frame to fix bugs are well framed	Time frames to fix bugs are not well framed and it varies depending on the nature of the bugs.
Team has freedom to decide what is necessary for the situation	Team may not have such freedom and constraint by the production system and end-user preference
Defects have no immediate effect on the production system	Defects may disrupt production system if not addressed properly
Developing a new system from the scratch is something unique	Maintenance activities are mostly repetitive in nature

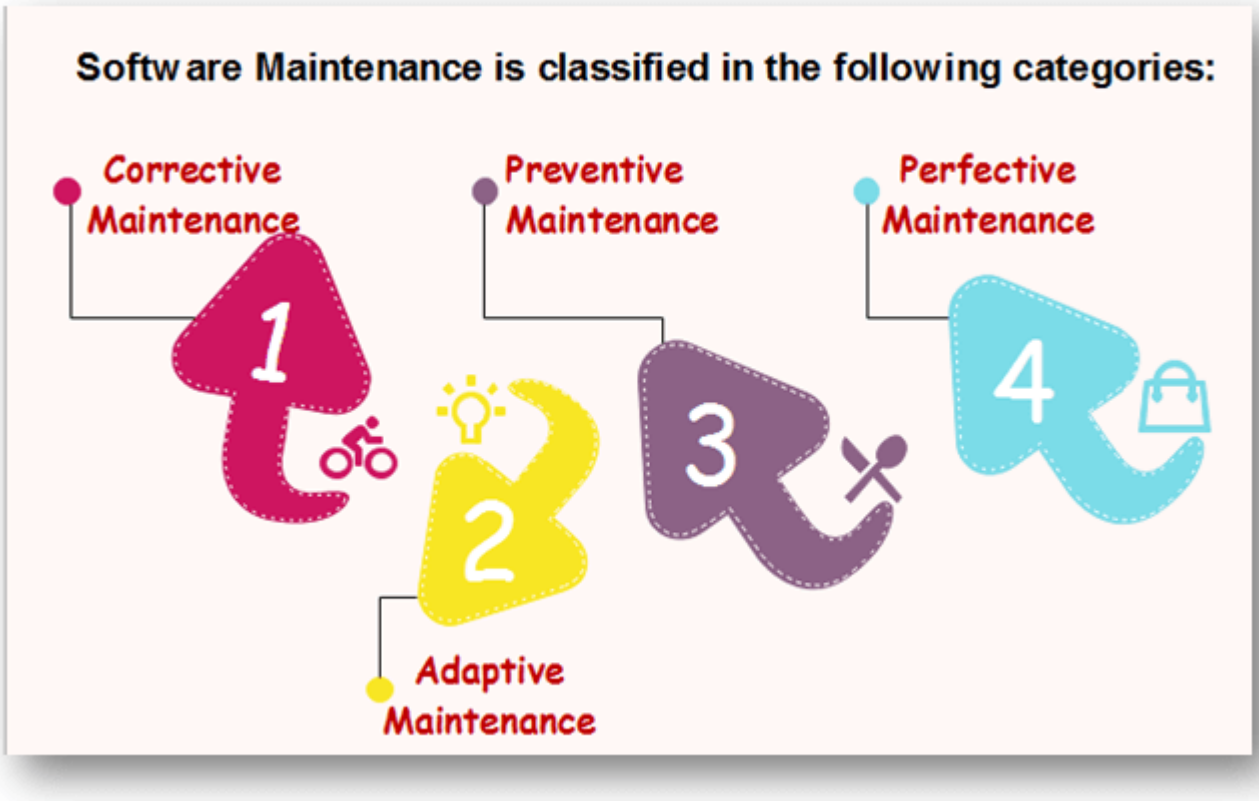
Need for Maintenance

Software Maintenance is needed for:-

- Correct errors
- Change in user requirement with time
- Changing hardware/software requirements
- To improve system efficiency
- To optimize the code to run faster
- To modify the components
- To reduce any unwanted side effects.

Thus the maintenance is required to ensure that the system continues to satisfy user requirements.

Types of Software Maintenance



1. Corrective Maintenance

Corrective maintenance aims to correct any remaining errors regardless of where they may cause specifications, design, coding, testing, and documentation, etc.

2. Adaptive Maintenance

It contains modifying the software to match changes in the ever-changing environment. Adjusting the system parameters as per the new hardware changes/os changes is also an example of adaptive maintenance. Mostly in adaptive maintenance, there will not be a change in functionality. Adding new features to the existing code.

3. Preventive Maintenance

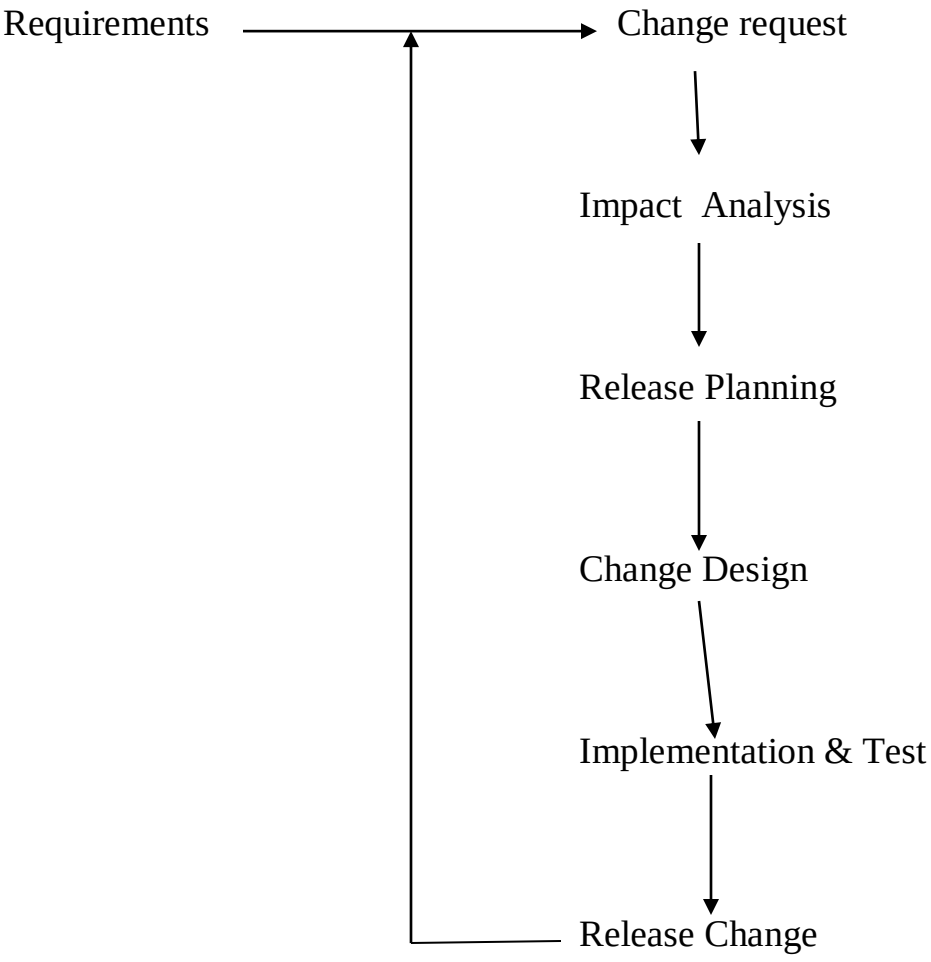
Activities that are aimed at increasing the maintainability are called as preventive maintenance. Documenting the existing system is the best example for preventive maintenance. Continuously monitoring the size of the database as the system grows to prevent sizing-related errors are examples of preventive maintenance. Only 4% of maintenance types are preventive maintenance.

4. Perfective Maintenance

It is the system upgrade kind of work which makes the system work more efficiently. About 50% of the maintenance is perfective maintenance. Upgrading hardware, software, and band widths are examples of perfective maintenance. Perfective maintenance happens due to adapting changes and due to user requirements. Typically made to improve the maintainability of code.

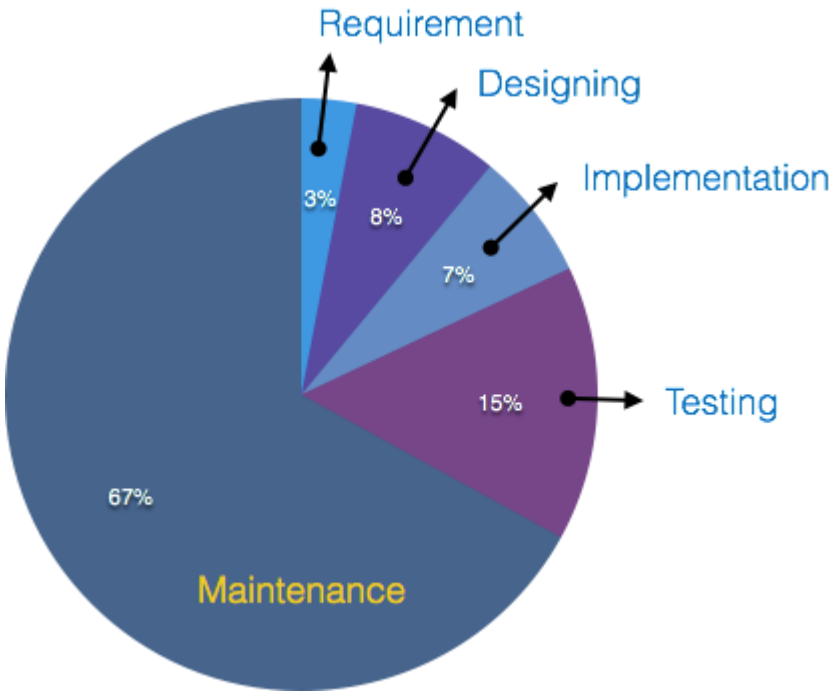
Maintenance Process

Maintenance processes vary considerably depending on the type of software being maintained. The most expensive part of software life cycle is software maintenance process.



Maintenance Cost

Maintenance cost is usually higher than the overall development cost of the software and sometime it is as high as 100 times the development cost.



There are various factors that determine the maintenance cost of the software, such as

1. **Complexity of the system:** When the complexity of the system to be maintained increases the maintenance cost also increases.

2. **Volume of work involved:** Sometimes the complexity may be less but the quantum of work involved in the maintenance activities may be high and hence the cost of maintenance also will be high.

3. **Aging of the software:** As the software age increases, the structure is degraded and hence it is harder to maintain and thereby increasing the overall maintenance cost of those systems.

4. **Soft skills:** Maintenance cost also involves various types of skills required to maintain the software. It not only depends on the quantity of the skills required, but also on the experience level of the people required to support the system.

5. **Contractual responsibility:** The maintenance cost also depends on the contractual responsibility of the vendor. If there is no incentive the vendor may not be interested for quick changes in the system.

Reverse Engineering

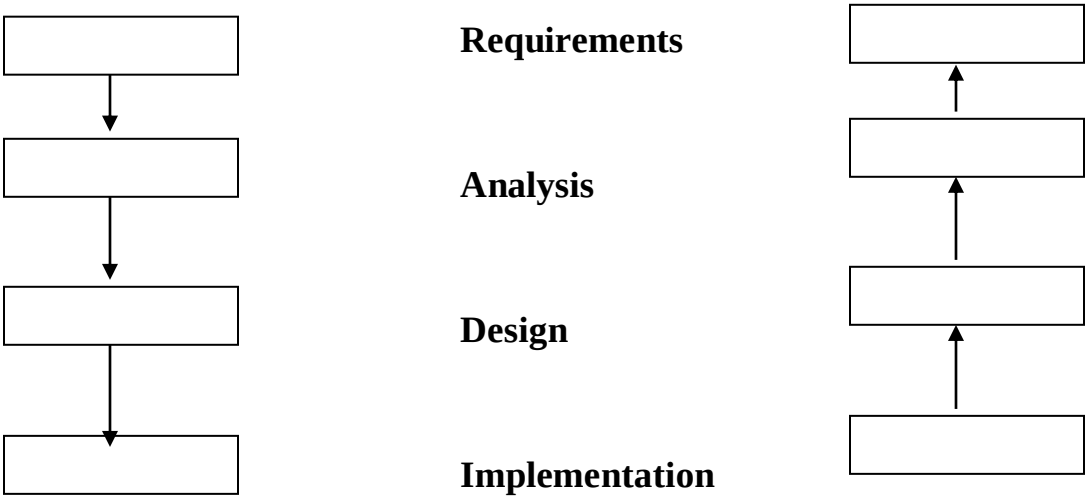
Forward Engineering:

In software development life cycle, we start from specification phase, analysis, design and implementation.

Reverse Engineering:

Process of creating of design document from source code and the specification document from design document.

Forward Engineering **Reverse Engineering**



- Analysis existing software with a view to understand its design and specification
- A process which analyses a product to find out the design aspects and its function.
- Builds a program database and generates information from this

Reverse Engineering Goals

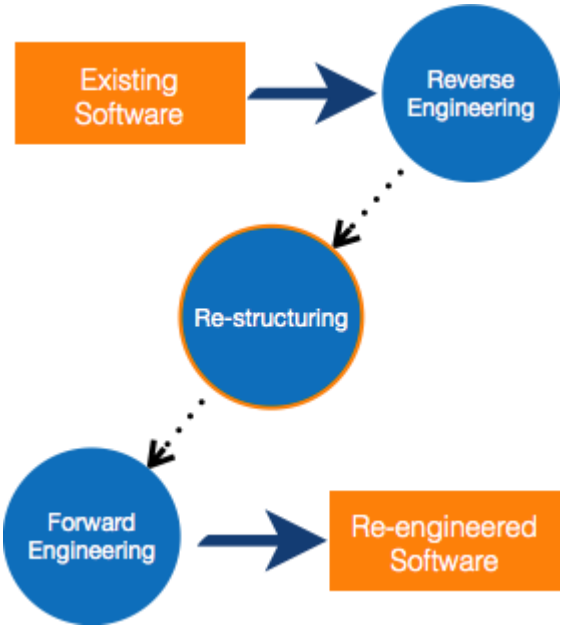
1. Cope with Complexity
2. Recover lost information
3. Detect side effects

Activities

- Understanding Process
- Understanding Data
- Understanding User Interfaces

Re-Engineering

Software Re-engineering is the examination and alteration of a system to reconstitute it in a new form. The principles of Re-engineering when applied to the software development process is called software re-engineering. It affects positively at software cost, quality, service to the customer and speed of delivery. In software re-engineering, we are improving the software to make it more efficient and effective.



Re-engineering may also lead to re-writing the software in entirely new programming language. Re-engineering may lead to re-documentation of the system. Re-engineering reduces the risks of the overall maintenance of the system. The cost of re-engineering is usually lesser than the cost of developing the new software.

Advantages of Re-engineering:

1. It reduces the risks: There may be high risks if we try developing new software; risks may be related to staffing problems, design problems, and specifications problems.
2. It reduces the cost: The cost of re-engineering is usually lesser than the cost of developing new software.

RE-STRUCTURING

Re-structuring is similar to re-engineering. Here the transformation of a system from one representation to another happens at the same level of abstraction. Functionality of the system does not change. Only code cleaning happens (clean code is code that is easy to understand and easy to change).

Migration from one version of the software to another version of the software is the best example of re-structuring.